REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 28/02/2010	**2. REPORT TYPE** Final	**3. DATES COVERED** *(From - To)* 12/01/2008-11/30-2009

4. TITLE AND SUBTITLE Adaptive Coordination for Dynamic Mobile Systems	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER FA9550-07-1-0157
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Christine Julien	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The University of Texas at Austin, One University Station, Austin, TX 78712	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Office of Scientific Research, 875 North Randolph Street, Suite 325, Rm 3112, Arlington, VA 22203	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFOSR
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-AFOSR-VA-TR-2016-0621

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution A - Approved for public release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
We have completed our work by finalizing the Evolving Tuples model and the Application Session middleware and connecting them in a complete middleware for prototyping mobile systems. Evolving tuples is an extension to traditional tuple spaces that allows applications to embed context-aware adaptation directly in structures traditionally used for distributed coordination. In the evolving tuples model, the tuples themselves can carry behavior specifications that define how they may move and change in response to the environment. The Application Sessions middleware provides a suite of metaphors for conversational semantics in dynamic environments. Applications can concretely define the semantics of their interactions and delegate the low level creation and maintenance of communication sessions to an underlying middleware. The Application Sessions model's explicit separation of concerns both architecturally and organizationally will lead to more robust, more successful, and ultimately more valuable software for mobile computing.

15. SUBJECT TERMS
middleware, mobile computing, dynamic systems, adaptation, software engineering

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*

FINAL TECHNICAL REPORT

Adaptive Coordination for Dynamic Mobile Systems

Reporting Period: 1 February 2007 to 30 November 2009

Grant No. FA9550-07-1-0157

Principal Investigator:
Christine Julien

University of Texas
Austin, TX

Adaptive Coordination for Dynamic Mobile Sytems

Dr. Christine Julien

Abstract

As ubiquitous computing applications become pervasive, the need for wireless, mobile coordination among dynamic participants is becoming a central focus of any application. Traditional distributed computing solutions require mobile parties to communicate to a central location (generally on an infrastructure network) before their data can be shared with other mobile parties. In contrast, ubiquitous computing environments benefit from direct communication among mobile parties that enables more efficient and speedy data sharing. Dynamic ubiquitous coordination environments abound in varying application domains. For example, an autonomously coordinating fleet of unmanned aerial vehicles requires significant direct interactions among the UAVs and perhaps among sensors on the ground. Future battlefields, construction sites, and repair zones will harbor large numbers of information-providing devices whose data can be shared for safety purposes or to make job functions easier or more efficient. Application development, however, is not the focus of this project. Instead, we identify what these applications have in common (i.e, dynamic environments, autonomy of devices, need for coordination, etc.) and encapsulate this shared functionality within a middleware infrastructure. This middleware will be supported by a a novel coordination model that handles the extreme dynamics these environments exhibit while leveraging the benefits of direct communication.

Research Objective. The goal of this research proposal is to create a coordination model and middleware for adaptive mobile applications. To enable adaptation, the coordination model specifically makes applications and the coordination among them *context-aware* by moving abstracted information about environmental state towards applications and makes coordination and communication *application-aware* by moving information about applications' constraints and requirements toward communication protocols. Ultimately, the work will produce a tailorable adaptive middleware that simplifies the programming of mobile applications without restricting the flexibility of coordination. This is not another middleware model proposal. The research plan involves neither simple tweaks to existing solutions nor a Herculean task of combining many layers into one massive solution. Instead, the project will devise a truly dynamic coordination model and middleware that adapt on-demand to changes in the environment and in application requirements or expectations.

Approach. In contrast to existing middleware approaches, our project does not focus in detail on rigid abstraction through layering. Instead, we focus on the information that can be shared among these abstractions through *cross-layer interactions.* We will pursue a top down approach that starts by creating programming constructs through which application developers can express their requirements for coordination. We will then create a dynamic coordination model that adapts to both application and context information. This coordination model includes a novel communication model that, by using context and application information, dynamically determines how messages should be routed. Modular implementations of the communication model can be swapped in, making the middleware itself tailorable to different operating conditions. Finally, we will also create adaptive sensing mechanisms that can tradeoff the fidelity of sensed context information for the overhead of sensing.

Scientific Merit. The novelty of this work in comparison with existing approaches is its dynamism based on interactions between layers of abstractions. This allows the resulting middleware to smoothly move information between traditional layers of abstraction and allows components within those layers to adapt their behavior to their environment.

Potential Impact. The project emphasizes the use of *awareness* to enable high-degrees of coordination among wirelessly connected, distributed components, thereby boosting the performance, efficiency, responsiveness, and flexibility in comparison to existing approaches. Such a solution will make possible significant amounts of coordination not previously possible in harsh network environments where operating conditions such as high degrees of mobility or low link quality have previously prevented reliable connectivity.

Adaptive Coordination for Dynamic Mobile Sytems

Research Effort

1 Introduction

This project targets the development and deployment of wireless applications in domains that require dynamic and adaptive coordination among highly mobile devices and devices embedded in the environment. Facilities for adaptive coordination are essential to making such applications responsive, efficient, and, most importantly, largely autonomous. Abstraction through *layering*, in which the problem is subdivided into multiple manageable tasks at each layer, is the most common and sensible approach to managing the complexity of this task. However, existing designs based on rigid abstractions ignore a significant amount of useful information that can be shared among layers for adaptation and increased performance. The impact of this information exchange can be enormous in providing an integrative platform for creating future computing systems. Such systems will necessarily be defined by distributed resources whose connectivity and coordination must be managed as an integral part of building any complete application.

Instead of focusing on the rigid separation of functionality into behavioral layers, this project will focus on developing novel ways of transiting information between components at different layers of abstraction. Specifically, the project provides *context-awareness* from the physical and network worlds and *application-awareness* from the logical application world. The former provides all components in a system (i.e., from communication protocols to applications) the ability to adapt their behavior to a changing situation. The latter (i.e., application-awareness) allows a system's coordination and communication to respond to changing application requirements. The project proposes new coordination models, algorithms, and protocols and throughout emphasizes the use of *awareness* to enable intelligent coordination among networked components, thereby boosting a system's autonomy, performance, efficiency, expressiveness, and flexibility in comparison to existing approaches.

We provide two applications that motivate the need for awareness at these varying levels. Within Section 2, we discuss not only these applications but also how such applications are addressed using current state-of-the-art technology. Section 3 overviews the project's intended solution strategy before Section 4 explores the project's specific research approach and intended results in more detail. Throughout, we compare our proposed approach with other existing endeavors. We conclude with a discussion of the potential impact of a successful project.

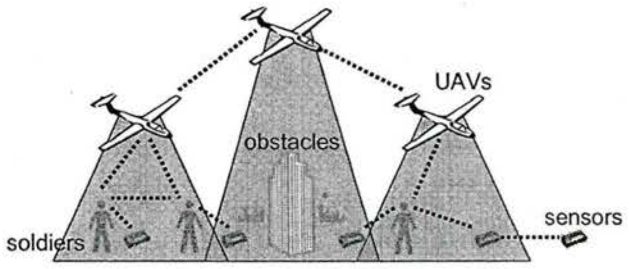

Figure 1: Coordinating UAVs, Soldiers, and Sensors

2 Motivating Scenarios

The first application scenario motivating this project involves enabling coordination among unmanned aerial vehicles (UAVs) that may be deployed for military, scientific, and civilian purposes, including border patrol, data gathering, reconnaissance, surveillance, and search and rescue missions. The coordination constructs this project will generate will ease the development of applications that require coordination among UAVs and vehicles, soldiers, agents, or sensors on the ground, as depicted in Figure 1. Such scenarios demand that the components automatically communicate, self-organize, collaborate, and undertake decisions autonomously, thereby adapting to achieve, for example, the best surveillance coverage within a designated area. The availability of information about coordinating partners and the quality and duration of communication with them is essential to planning in this type of application. The result of such application support would be a coordinated, intelligent, and largely autonomous fleet of UAVs. With current technologies, UAVs deployed for any task are almost entirely operator controlled. In addition, moving information from one UAV to another is not done directly; information must first be transmitted to a ground station where it is processed by an operator before it can impact the behavior of another UAV (through explicit operator control). Direct communication among the UAVs (potentially augmented with information from sensors in the environment) enables more rapid response to changing situations. Active coordination using this direct communication makes the UAV fleet more autonomous and adaptive. In addition, minimizing the required physical communication using direct connections between mobile and embedded devices can help relieve the over-taxed communication links present in today's military wireless networks.

Our second motivating scenario enables real-time coordination and collaboration of workers in a dynamic information environment. This scenario envisions a set of workers distributed around a hangar performing maintenance on a variety of aircraft systems across many aircraft. Both equipment and information must be shared among maintainers in this environment. The latter includes information collected automatically by an airplane during its mission, repairs required or requested for a particular craft, the status of repairs, an aircraft's maintenance history, etc. Equipment shared among the maintainers includes not only the aircraft but also expensive equipment within the hangar used for both diagnostics and repair. In addition, direct wireless connectivity from the floor of the hangar to the parts storage and ordering facilities can increase the speed and efficiency of repairs. Finally, maintenance officers oversee the progress of all repairs and must sign off on complete repairs. With current technologies, maintenance activities are minimally aided by computing technologies through a laptop-like interface that can store digital copies of maintenance records and manuals. This interface provides only asynchronous (or *off-line*) collaboration, achieved through docking the laptop at certain times or at the end of the day. The coordination constructs this project proposes would augment existing capabilities with real-time collaboration among dynamic parties, allowing maintainers to keep a consistent picture of the status of on-going repairs and availability of other personnel, equipment, and parts. This application is similar in flavor to the vision of the intelligent construction site [13], an area in which the PI and her collaborators have already developed a prototype coordination middleware [16, 17]. The similarity between these domains offers a unique opportunity to test the results of this proposed project in a similar yet less mission critical environment prior to deployment.

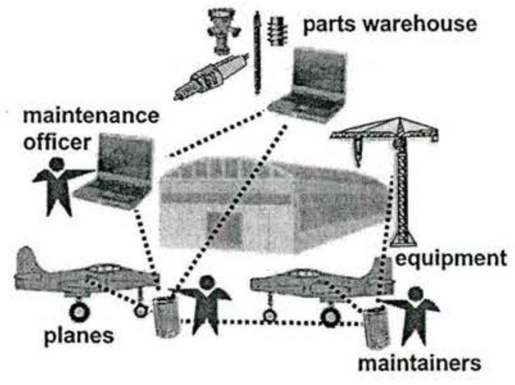

Figure 2: A Coordinated Maintenance Environment

3 Overview of Solution Strategy

While the above applications provide motivation for this project's research, the ultimate goal is not to develop these applications but to make it possible and simple to create such dynamic applications using efficient and intuitive programming constructs. Specifically, this project will provide adaptive coordination through precise application-awareness and efficient context-awareness. The project will result in a middleware that will provide a programming interface to application developers that enables the rapid development of highly-adaptive applications.

In the model and system this project will create, awareness moves information about the application and the context through three hierarchical layers: 1) *application coordination*, the major interface to the application, where application requirements are collected and high-level coordination occurs; 2) *communication*, through which tailored wireless communication protocols enable intelligent creation and maintenance of groups of coordinating devices; and 3) *context collection*, which efficiently and transparently collects and abstracts the context information dynamically deemed to be necessary or beneficial. The work proposed in this project complements an ongoing collaboration with Dr. Sriram Vishwanath (see PRINCIPAL INVESTIGATOR TIME, current project *CSR–SMI: ACLI-ware*) that aims to use cross-layer interactions to allow information about physical wireless channel state to impact applications and to allow applications to impact how wireless channel state is sensed. In contrast, this proposed project focuses specifically on the coordination gains that can be achieved when one has access to context information and how coordination constructs can provide application preferences to underlying system components.

Section 4 of this proposal describes the specific intended research contributions. In the remainder of this section, we first discuss the types of application and context information that can be shared among traditional system layers to improve coordination autonomy and efficiency.

3.1 Application-awareness

Interactions that fall under the guise of *application-awareness* distribute information about application behavior and requirements to the underlying model and middleware components.

- *Application-awareness for application coordination*: to be useful to applications, the lower-layer model and middleware components must provide a usable interface for developers to express their constraints and thereby influence the middleware's behavior. The project will create expressive coordination constructs that allow applications to define policies related to their expectations and allowable tradeoffs. This information can enable adaptive coordination through dynamic selection of coordinating partners. This work will build on my existing work in using traditional coordination languages for expressing such policy parameters.

- *Application/coordination-awareness for communication*: applications' reliability requirements and preferences can also be used to determine the best communication links and routing paths to use to maintain connectivity of a coordinating pair or group. The clustering models and protocols this project will develop will use application-provided information to intelligently adapt group formation and communication protocols to provide efficient and responsive implementations that exactly match applications' requirements without incurring more than the necessary communication overhead.

- *Application/communication-awareness for context sensing*: information about the physical and network environments can be sensed at varying levels of granularity. One can devote a lot of energy to determining context information to a high degree of accuracy only for the information to provide little to no benefit at higher layers. In addition, context sensing can be very expensive because it generates added communication overhead. We will devise context sensing techniques that respond directly to application requirements to collect only context information that applications or communication protocols can use while incurring as little additional overhead as possible.

3.2 Context-awareness

In the opposite direction from application-awareness, all three levels can also use information about the physical and network environments to adapt their coordination behavior.

- *Context-awareness for context sensing*: at first glance it may seem strange that the context sensing itself should adapt, but information sensed, for example, about the network conditions can tell a context sensing algorithm to use strictly *passive sensing* (when network activity is high) or to use a partially active approach (because little other communication is occurring). This project will model the use of such context information and will create context sensing protocols that can specifically respond to such information.

- *Context-awareness for communication*: group formation and communication algorithms can benefit significantly from information regarding what other devices are available. In addition, information about the physical world (e.g., how fast devices are moving) and the network world (e.g., how congested the network is) can help select the most efficient communication approach for a set of operating conditions. This work will define an infrastructure that allows effective communication of exactly the context information required by communication protocols. In addition, we will create a communication suite that adapts itself to these changing conditions to provide reliable connectivity even in harsh environments.

- *Context-awareness for application coordination*: finally, applications can use context information to determine what information to send when. In dynamic networks, connectivity must be leveraged opportunistically because there is no guarantee of persistent connectivity. When a network is congested or the link quality is low, only high priority information should be transmitted. Applications can also adjust their *data fidelity* in response to context (e.g., to send low quality video when only low bandwidth links are available). We will develop a language to communicate context and connectivity information to applications and a framework within our middleware that applications can use to respond to such context information.

3.3 Benefits of the Approach

By undertaking a solution strategy founded on bi-directional interactions among physical and network components and application components, this project will provide several benefits over related approaches that investigate either only protocols within single layers of abstractions or approaches that create awareness in only one direction. First, as demonstrated by the vast differences in our motivating applications, this project's broader, integrative perspective unifies disparate applications within a single system, making system maintenance and updating much simpler. In addition, the resulting model's ease of understanding and the system's ease of use encourage rapid adoption and deployment of cutting edge applications that can share a single underlying system infrastructure.

4 Research Approach and Nature of Expected Results

As intimated above, we divide the research tasks into two areas; concerns related to creating a complete and correct model of the ways applications should interact and share information across traditional protocol layers and concerns related to providing a practical realization of these models in a dynamic middleware. Instead of handling modeling and systems issues as separate tasks, we integrate both our discussion and intended investigation of them to ensure both that our system realization is grounded in formal underpinnings and that our model is in fact implementable. Figure 3 shows the areas in which the research contributions described below fall. We describe these areas from the top down, focusing on the provision of application- and context-awareness depicted to the left and right sides of this figure instead of on the traditional requests and replies found between hierarchical layers. These aspects of awareness differentiate this project from other mobile coordination efforts. At the top of the figure, high-level specifications of interactions provide an intuitive interface to application programmers, allow applications to provide information to both coordination and communication constructs (on the left), and allow applications to respond to context information (on the right). In the second step, we define a coordination model that is largely hidden from the application developer (through the high-level specification interface) but provides an important abstraction of the underlying physical world. The coordination model adapts to information about the environment and network states and provides awareness of coordination requirements to communication protocols through a novel encapsulation of communication behavior. The communication model provides a standard manner of supporting coordination, and it is realized by concrete implementations of communication protocols. While these communication protocols can adapt to application requirements through information fed from the coordination model, they also provide valuable information about the state of the network and thereby enable context-awareness for the upper layers. Finally, an environmental sensing module augments the network awareness to provide complete context-awareness.

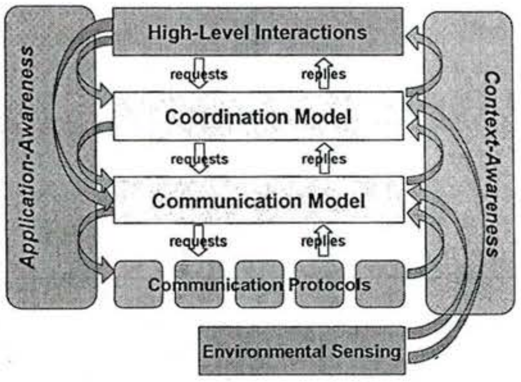

Figure 3: Layering and Awareness

4.1 High-Level Specifications of Interactions: Enabling Application-Awareness

At the level closest to the application developer who will use our integrated coordination system, we will provide constructs that enable high-level specifications of interactions among dynamic mobile devices This portion of the proposed work is founded on our existing *application sessions model* [23] which formalizes a suite of interactions among devices in a mobile computing environment. These interactions range from short-lived interactions among two parties to long-lived interactions coordinating a group of participating entities. By its construction, the model explicitly separates the user program (i.e., the local behavior of a particular device) from interactive session management, which the application can then delegate to an underlying coordination system. The only information shared between the two are a specification (*spec*) of the desired coordinating partner or information, and a handle (p) that allows the application to access the resource to which the underlying infrastructure connects it. To date, this application sessions model includes four types of sessions. The *query session* exemplifies a simple, one-time request for a piece of data from a remote party. The *provider session* connects the application to a particular remote device satisfying some requirement. The infrastructure subsequently maintains this connection even in the face of mobility or other dynamics. The *type session* connects the application to a remote device that satisfies some requirement. As mobility or other dynamics cause changes in the environment, the application can be reconnected to a different coordinating partner that also satisfies the initial requirement. This session type provides more flexibility within the implementation of the underlying coordination model. Finally, the *group session* connects the requester to all communicating parties that satisfy some requirement (e.g., UAVs within a certain area). As dynamics cause this set to change, the underlying coordination infrastructure updates the participants in the group session. For brevity, we show the formalization of only the *type session* here; complete formalizations can be found in [23]:

```
$\boxed{p \Leftarrow spec}$
    $\triangleq$  $p = p'.(p' \models spec \wedge p'.\texttt{connected})$
        while $p \neq \epsilon$ do
            $\langle$await $\neg p.\texttt{connected} \rightarrow p = p'.(p' \models spec \wedge p'.\texttt{connected})\rangle$
        od
```

The expression in the box denotes the particular session semantic, in this case the type session is indicated using an open arrow to assign the specification *spec* to the handle p. The selection of a coordinating partner matching the requirement uses *non-deterministic assignment* [2] to indicate that a partner is selected from any that satisfy the specification. A statement $x := x'.Q$ assigns to x a value x' nondeterministically selected from among the values satisfying the predicate Q. If such an assignment is not possible, the statement aborts; we assume this results in assigning ϵ (a null value) to x. The entails ($\models$) relation indicates that a remote device satisfies a specification, i.e., in $p \models spec$, remote device p satisfies *spec*. In the type session specification above, the requester is initially connected to any remote device that satisfies the specification and is connected to the requester. If this coordinating partner becomes disconnected, the type session attempts to connect to any other remote device that happens to be connected and satisfy the specification.

Within this proposed project, we will use this existing work as a foundation for creating session specifications that communicate more than just functional requirements from the application to the underlying coordination infrastructure and communication protocols. These new constructs will provide much of the information that our vision of *application-awareness* requires. We envision augmenting application's specifications of interactions with a *preference function* that can encapsulate these non-functional requirements. For example, a type session's specification may become:

```
$\boxed{p \Leftarrow spec/f}$
    $\triangleq$  $p = p'.(p' \models spec \wedge p'.\texttt{connected} \wedge \langle \forall \pi : \pi \models spec :: f(p') < f(\pi) \rangle)$
        while $p \neq \epsilon$ do
            $\langle$await $\neg p.\texttt{connected} \vee \langle \exists \pi : \pi.\texttt{connected} \wedge \pi \models spec \wedge f(\pi) < f(p) \rangle \rightarrow$
                $p = p'.(p' \models spec \wedge p'.\texttt{connected} \wedge \langle \forall \pi : \pi \models spec :: f(p') < f(\pi) \rangle)\rangle$
        od
```

In this case, f is a function that compares the quality of two matching potential partners. Not only does the type session have to ensure that it initially selects the best partner, but it must constantly monitor the available potential partners to ensure that this match remains the best one available. The statement inside the loop ensures that the requester is reconnected if a better provider becomes available.

In the above example, the function f makes the process of incorporating preferences simple, but several subtleties arise in how the function f should be written and how it should be applied in all cases. In addition, incorporating this constant monitoring of potential partners during a long-lived interaction changes how the coordination infrastructure can successfully implement these interactions, as described in more detail below. Within this project, we will investigate how to use this preference function approach to enable novel application semantics (e.g., a probabilistic group session that connects the requester to some percentage of the devices that satisfy the requirement or a type session that allows the requester to dictate conditions under which a binding can be reassigned).

Our approach is similar in spirit to approaches that perform network sensitive service selection [18] or approaches that use Quality of Service (QoS) to differentiate potential service providers [25, 26, 32]. An important distinction of our design, however, is the placement of this function in the hands of the requester and the complete distribution of its evaluation. Through the combination of our specification mechanics and the implementation of the evaluation (described below), we provide a novel evaluation structure that functions without reliance on any central service or third party to assist in the process. In addition, our approach to constructing f will focus on ensuring the simplicity of its specification, a concern which many of the approaches above ignore due to the fact that policies are fairly statically defined.

Ultimately, our work in this thread will result in a usable interface for application developers to easily express their constraints and tradeoffs with respect to coordination interactions in a dynamic environment. This approach provides application-awareness by allowing applications to place constraints on lower layers. It also allows the application to express how context information impacts the application layer and therefore dictates what context information should be collected and with what fidelity.

- **Research Task #1**: Creation of application sessions that allow different semantics, encapsulate the requester's non-functional requirements into the formalizations of the interactions, and provide intuitive programming constructs on top of the underlying coordination middleware.

4.2 Supporting High-Level Interactions through a Novel Coordination Model

In this section, we discuss the research issues involved in creating a complete coordination model to underly the high-level interactions specified through the constructs described above. This new model will be founded on Linda-like tuple spaces [9, 14] that enable content based coordination among distributed parties. We combine the notion of a Linda tuple space with the global virtual data structures model [29] in a manner similar to LIME [28] and EgoSpaces [22] by associating a local tuple space with each mobile device in the network. This tuple space contains tuples with the information a particular device shares with other networked devices. A global virtual tuple space is dynamically defined to contain the contents of all tuple spaces of devices that are simultaneously connected. It is this dynamic global virtual tuple space, instead of a single, centralized one, over which interactions in our coordination model operate. Figure 4 shows how this global virtual data structure abstraction dynamically creates coordination groups based on connectivity.

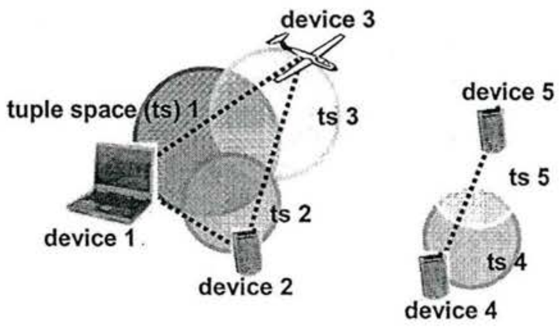

Figure 4: Dynamic coordination groups based on global virtual data structures. Devices 1, 2, and 3 form a group; devices 4 and 5 form a second group.

Information about devices in the network and the data they create is stored in the device's local tuple space in *tuples*. Each device may have a single tuple that describes itself and may produce more tuples that change over time as the data it collects or the services it can offer change. From our first motivating example, each UAV may contain a tuple that describes the region on the ground that the UAV can currently observe. A group session defined over a set of connected UAVs could generate a complete picture of the currently observed ground area. From our second example, the following tuple stored in an aircraft's local tuple space may indicate an anomalous event that occurred during that aircraft's mission:

$$\langle (\textit{event}, \text{system failure}), (\textit{system}, \text{avionics}), \ldots \rangle$$

Each pair in the tuple is a *field* and contains a portion of the tuple's information. In this case, the ellipses indicates that the event information may be augmented with additional details that pertain to the particular failure (e.g., the time of the incident, particular components implicated, etc.). At the lowest level, requests for data take the form of *templates* over tuples that specify the requirements for a matching tuple. For example, an avionics maintainer in the hangar may search for avionics mission events using the following template, which requires a matching tuple to have, at a minimum, two fields that satisfy the stated restrictions:

$$\langle (event, *), (system, = \text{avionics}) \rangle$$

This template matches any event that occurred in the avionics systems, including the tuple shown above.

This use of semi-structured data [1] to describe data and requests for data is not novel to our particular approach and has been used in service description languages [4, 10, 12] and other tuple-based systems [8, 20, 28]. What is novel about our approach is how we incorporate the application-awareness made possible by the interaction semantics described above into the tuple-based coordination model. In our model, interaction requests generated by an application's use of the high-level interaction semantics above are encapsulated in *active tuples*, similar to those first introduced in Linda [9]. An active tuple differs from the previously described passive tuples in that it can contain uncompleted computation. In our new coordination model, this computation is specifically the series of steps that, in conjunction with a communication protocol, move the active tuple to the appropriate location (i.e., where the desired information resides), evaluate the request against available coordinating partners, and send a result back to the requester. The active tuple is also used to transit application information to the coordination infrastructure and communication protocols. In general, an active tuple contains some unevaluated aspect when it is placed in the requester's local tuple space. The evaluation of the active tuple is performed as it is moved by the communication model and ultimately resolved to connect the requester to a remote coordinating party. The final component of our new coordination model is this modular communication model that causes movement of active tuples, resulting in their successful evaluation and the return of responses to requests. Together the above components constitute a complete coordination model that directly supports the framework's interactive constructs described in the previous section. Understanding and incorporating these individual components into a complete model is a research task in its own right; in the next two sections we investigate the use of active tuples and modular communication in more detail.

4.3 Incorporating Application- and Context-Awareness into Active Tuple Evaluation

The functional requirements for a coordination interaction (i.e., properties of the coordinating partner or information to be shared) are captured in the interaction's semi-structured data specification which the communication model described below moves through the network until it finds a match. Non-functional requirements such as application- or context-awareness are instead captured within the active tuple itself. To accomplish this, part of the unevaluated portion of the active tuple instructs the communication protocol on how to forward the active tuple through the network based on application preferences or the state of the network or physical environment. Specifically, it is at this point that the coordination model takes advantage of the application information encapsulated in the high-level interaction specification through the preference specification introduced as part of **Research Task #1**.

Within the scope of application-awareness, not only should information from the application requesting the coordination impact the evaluation, but this process must also be aware of properties of potential coordinating partners' applications. If a request finds multiple potential partners that could satisfy the request, aspects of the potential partner should be used to help determine which is the best option. For example, if a maintainer is searching for a particular piece of equipment to use for a repair, the request should prefer equipment that is not already in use for another repair. From the perspective of the requester, aspects of the application that may impact the process of selecting coordinating partners might include aspects such as the proximity of the coordinating partner (i.e., location-dependent interactions tend to favor closer partners) or the coordinating partner's relative mobility (e.g., if a UAV needs to coordinate with one other nearby UAV, it may prefer the stable connection to a UAV moving in the same direction at the same speed). In addition, the potential partner may have requirements that a request must meet before the partner will coordinate. Before accepting a coordination interaction, a potential partner may evaluate its current battery power (since communication expends valuable energy) or its load (i.e., the number of other coordination connections it is already supporting).

Finally, because applications like those described in Section 2 are supported by mobile ad hoc networks in which the nodes themselves serve as routers for connections between devices that are not directly connected, the process of moving active tuples towards potential coordinating partners should also account for the state of the intermediate network. That is, the way coordination occurs should also be *context-aware*. For example, the latency of the end-to-end network connection, the number of network hops required to connect coordinating partners, the battery power available on the intermediate nodes to support the connection, etc., should all factor into whether a particular coordination partnership is established. From an implementation perspective, this is accomplished by hooking the coordination model into the underlying context-awareness provisions through active tuple evaluations. When active tuples reach an unevaluated portion of code that requires context information, the context-sensing components (discussed in Section 4.5) kick into gear to generate and return the required information. The result can determine, for example, whether or not the communication protocol continues to forward the particular active tuple. This movement of active tuples to support coordination has been explored in systems in addition to the original Linda model. X-KLAIM [5] used active tuples to explicitly code movement of mobile agents from one named location to another. Closer to our work, TOTA [27] enabled active tuples to themselves carry communication protocol information that determines how they should be propagated. We explicitly separate the communication modules from the coordination components to create a more modular and adaptive system. In our system, the active tuples influence the routing protocol based on application- and context-awareness but do not explicitly control communication..

To satisfy all of these concerns, we will divide the preference function introduced in Section 4.1 into three partial functions: $f_a(p)$, which defines the cost to the application requester a of selecting a particular partner p; $f_p(a)$, which defines the cost to partner p of coordinating with the requesting application a; and $f_n(a,p)$, which defines the cost to the network to support the coordination between a and p. In combination, the overall preference function f from Section 4.1 is:

$$f(a,p)^1 = \alpha f_a(p) + \rho f_p(a) + \nu f_n(a,p)$$

where α, ρ, and ν are system-specified constants that weight the input of different participants. The coordinating partner that best satisfies this function at any given instant has a cost of:

$$c_{optimal} = \langle \min a, p : p \models spec :: f(a,p) \rangle^2$$

All of the partial cost functions ($f_a(p)$, $f_p(a)$, and $f_n(a,p)$) return values between 0 and 1, and $\alpha + \rho + \nu = 1$, so that the result of evaluating the global preference function is a value between 0 and 1. Each of the partial cost functions can also return ∞, which effectively vetoes the particular coordination interaction.

As part of this proposed work, these pieces of the preference function must be incorporated into the active tuples that carry and evaluate requests for coordination connections. Our initial work in this area [19] serves as a starting point for injecting our coordination model with application- and context-awareness by representing these functions as unevaluated portions of active tuples. As an active tuple carrying a request is moved through the network by the communication protocol described next, these functions will be incrementally evaluated. The incremental evaluation incorporates both application-awareness (where information comes from the requester's specified preferences and information about the potential partner's application available in its local tuple space) and context-awareness (where the information comes from passive sensing from communication protocols and hybrid passive/active sensing from environmental sensors, as described below).

- **Research Task #2**: Representation of application- and context-awareness as unevaluated portions of active tuples that are incrementally evaluated as requests are moved through a network.

[1]In Section 4.1, we wrote this function as $f(p)$ because it is written by a particular application from its perspective, and a is therefore implied. Here, for completeness, we include a in the function.

[2]In the *three-part notation*: $\langle$**op** *quantified_variables* : *range* :: *expression*$\rangle$, the variables from *quantified_variables* take on all possible values permitted by *range*. Each instantiation of the variables is substituted in *expression*, producing a multiset of values to which **op** is applied, yielding the value of the three-part expression. If no instantiation of the variables satisfies *range*, then the value of the three-part expression is the identity element for **op**, e.g., *true* of **op** is $\forall$ or 0 when **op** is min.

4.4 Modularizing Adaptive Communication

To successfully resolve interaction requests contained in active tuples, the coordination model must enable communication among requesters and potential coordinating partners. In our coordination model, we accomplish this by moving the active tuples containing the requests. In the highly dynamic mobile environments that characterize our applications, this process cannot rely on a centralized server to resolve requests because coordination groups must be formed by dynamically available coordinating partners, as shown in Figure 4. Instead, we rely on techniques for *mobile ad hoc networks* that form opportunistically given available communicating partners and change rapidly in response to mobility. In such networks, mobile devices must serve as routers for connections among other parties, e.g., in a network of coordinating UAVs, the UAVs may not all be directly connected to one another, and end-to-end connections may have to be partially supported by other UAVs (as shown in Figure 1).

Our *communication model* specifies the interface between communication and coordination. This interface defines the specific functionality that any communication protocol must provide and enables our approach to be highly modular, allowing one communication paradigm to be swapped out for another. This can be very important in mobile computing environments where the degree of mobility or other change can radically impact which communication approach provides the best performance [6, 7, 11].

The communication model receives requests for communication by receiving active tuples from the coordination model. Part of the unevaluated portion of the active tuple contains the specification of the desired interaction partner (*spec* from Section 4.1 above). When the active tuple encounters a coordinating partner that matches this specification, this portion of the active tuple will evaluate, resulting in a "match" for the request. This matching process is augmented by the incremental evaluation of the preference functions described in the previous section which may further restrict which partners match. The communication model itself dictates that such requests must be accepted by any communication protocol and that, when matches occur, a reply is generated and transmitted back to the requester. Depending on the degree of application-awareness, the coordination model may collect the replies before determining the "best" match to deliver to the application. Communication protocols that fit within the communication model may perform this function in different ways appropriate to their particular situations. The research required for the communication model includes formalizing how its constructs fit with the coordination model and what happens when active tuples move from one device to another.

In addition, to provide a system realization of our integrated coordination model, we will create a suite of communication protocols that adhere to this model and successfully move active tuples to resolve application requests. The first of these communication approaches, *Cross-Layer Discovery and Routing* (CDR) [24], is a simple constrained flooding based protocol that uses the application request itself to help determine which communication paths to use. CDR is a completely reactive routing protocol that performs communication requests only on demand. In addition, CDR does not currently use any context information to help determine which communication path is the instantaneous best choice. In building a suite of modular protocols, we will build on CDR to create additional options that perform a hybrid of reactive and proactive routing and use context information to adapt the routing decisions. Group-based interactions generally require a different style of interaction entirely. Communication protocols for supporting group communication will be founded on our Network Abstractions protocol [21, 30] which encapsulates a context-aware multicast protocol. We will also explore other paradigms of communication that may prove viable in our dynamic networks, perhaps based on overlay routing or cluster based communication.

- **Research Task #3**: Definition of a model of modular context- and application-aware communication that ultimately handles the distribution of requests for coordinating partners and data and the funneling of replies and information back to requesters.
- **Research Task #4**: Creation of a suite of modular communication protocols that adhere to the model defined in **Research Task #3** but respond in different ways to changing situations to produce context-aware communication.

4.5 Adaptive Context Sensing

Application-awareness is enabled by the gathering of preferences from requesters and potential coordinating partners as detailed in the previous sections. The communication protocols and active tuple coordination strategy both rely on the availability of context information to enable communication paths and coordination interactions to adapt to a dynamic environment. Within our model, context-awareness comes in two forms: *environmental awareness*, or knowledge about aspects of the physical world; and *network-awareness*, or knowledge about the current state of communication links. In both cases, actively collecting information about the environment can be very expensive as it commonly generates extra communication, which can cause network overhead and expend valuable battery power.

We define passive sensing as the use of eavesdropping on communication or other network or environmental tasks to infer context measurements. Our previous work has shown that we can effectively and efficiently estimate a local degree of mobility in such a manner [31]. Other passive sensing efforts have used incoming radio signal strength [3] or TCP packet traffic information [33] to adapt local protocols to changing network conditions. Work under the umbrella of this proposal will complete a suite of such metrics that interact with available communication protocols and environmental sensors to create highly passive metrics. These metrics will be combined with an environmental sensing package such as [15] to provide hybrid metrics that perform as much passive sensing as possible but are able to trade off overhead for higher fidelity context information. This tradeoff is at least partially dictated by information about the degree of context-awareness that applications require, as specified in the high-level interactions specifications. Our complete context sensing package will be incorporated into the framework as shown in Figure 3. In addition to defining the sensing suite, this research will also provide generic hooks into and out of the context sensing package to allow it to connect to the communication and coordination models to receive application-level information about allowable tradeoffs and to distribute sensed context information into the coordination and communication protocols so they can adapt as described in the previous sections.

- **Research Task #5**: Creation of a suite of sensing protocols that provide a combination of passively and actively sensed context information at a very low overhead.

5 Potential Impact

The research tasks embedded in this project promise to introduce the benefits of combining *application-awareness* and *context-awareness* to create highly adaptive coordinating applications. This starts, first and foremost, with an expressive but simple programming interface presented to the application programmer. Through this interface, the programmer can enable application-awareness for the lower levels and dictate the kinds of context-awareness that will benefit it directly. Behind this programming interface, a novel coordination model provides the infrastructure over which applications' high-level interactions can be performed. Two key aspects of this coordination model, *active tuples* and the *modular communication model* create a highly modifiable and adaptive implementation of the coordination tasks. Finally, intelligent sensing of network and environmental context provides much needed information to dynamically tailor the behavior of the system from the communication protocols up through the application.

While this project does propose novel algorithms, models, and coordination techniques, it emphasizes the use of *awareness* to enable novel interactions among these components, thereby boosting the performance, efficiency, responsiveness, and flexibility in comparison to existing approaches. If successful, such an approach will make possible significant amounts of coordination not previously possible in harsh network environments where operating conditions such as high degrees of mobility or low link quality have previously prevented reliable connectivity. By sensing and adapting to context information, the middleware developed by this project will tailor application support to particular operating conditions, creating a flexible and responsive coordination substrate that changes as the physical and network environments change.

References

[1] S. Abiteboul. Querying semi-structured data. In *Proceedings of the 6th International Conference on Database Theory*, pages 1–18, January 1997.

[2] R.J.R. Back and K. Sere. Stepwise refinement of parallel algorithms. *Science of Computer Programming*, 13(2-3):133–180, 1990.

[3] P. Basu, N. Khan, and T. Little. A mobility based metric for clustering in mobile ad hoc networks. In *Proceedings of the International Conference on Distributed Computing Workshops*, pages 413–418, 2001.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[5] L. Bettini, R. De Nicola, R. Pugliese, and G. Ferrari. Interactive mobile agents in X-KLAIM. In *Proceedings of the 7th International Workshop on Infrastructure for Collaborative Enterprises*, pages 110–115, June 1998.

[6] A. Boukerche. Performance evaluation of routing protocols for ad hoc wireless networks. *Mobile Networks and Applications*, 9(4):333–342, 2004.

[7] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the ACM International Conference on Mobile Computing and Networking*, pages 85–97, 1998.

[8] G. Cabri, L. Leonardi, and F. Zambonelli. MARS: A programmable coordination architecture for mobile agents. *IEEE Internet Computing*, 4(4):26–35, 2000.

[9] N. Carriero and D. Gelernter. Linda in context. *Communications of the ACM*, 32(4):444–458, 1989.

[10] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web services description language (WSDL) 1.1, 2001. Current as of 2005.

[11] S.R. Das, C.E. Perkins, and E.M. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proceedings of the Conference on Computer Communications*, pages 3–12, 2000.

[12] K. Edwards. *Core Jini*. Prentice Hall, 1999.

[13] Fiatech. Strategic overview: Capital projects technology roadmap initiative (version 1). `http://www.fiatech.org/projects/roadmap/cptri.htm`, 2003.

[14] D. Gelernter. Generative communication in Linda. *ACM Transactions on Programming Languages and Systems*, 7(1):80–112, 1985.

[15] G. Hackmann, C. Julien, J. Payton, and G.-C. Roman. Supporting generalized context interactions. In *Software Engineering and Middleware: 4th International Workshop, Revised Selected Papers*, volume 3437 of *Lecture Notes in Computer Science*, pages 91–106, 2005.

[16] J. Hammer, I. Hassan, C. Julien, S. Kabadayi, W. J. O'Brien, and J. Trujillo. Dynamic decision support in direct-access sensor networks. submitted to *the 3rd IEEE International Conference on Mobile Ad-hoc and Sensor Systems, Demonstrations*, 2006.

[17] J. Hammer, C. Julien, and W. J. O'Brien. Minimal footprint decision support in sensor networks. submitted to *the 25th Army Science Conference*, 2006.

[18] A.-C. Huang and P. Steenkiste. Network-sensitive service discovery. *Journal of Grid Computing*, 1(3):309–326, 2003.

[19] C. Julien. Adaptive preference specifications for application sessions. Technical Report TR-UTEDGE-2006-006, The Center for Excellence in Distributed Global Environments, The University of Texas at Austin, 2006.

[20] C. Julien and G.-C. Roman. Egocentric context-aware programming in ad hoc mobile environments. In *Proceedings of the 10th International Symposium on the Foundations of Software Engineering*, pages 21–30, November 2002.

[21] C. Julien and G.-C. Roman. Supporting context-aware interaction in dynamic multi-agent systems (invited paper). In *Environments for Multi-Agent Systems*, volume 3374 of *Lecture Notes in Computer Science*, pages 168–189, February 2005.

[22] C. Julien and G.-C. Roman. Egospaces: Facilitating rapid development of context-aware mobile applications. *IEEE Transactions on Software Engineering*, 32(5):281–298, May 2006.

[23] C. Julien and D. Stovall. Enabling ubiquitous coordination using application sessions. In *Proceedings of the 8th International Conference on Coordination Models and Languages*, pages 130–144, June 2006.

[24] C. Julien and M. Venkataraman. Cross-layer discovery and routing in reconfigurable wireless networks. In *Proceedings of the 3rd International Conference on Mobile Ad-hoc and Sensor Systems*, 2006. (to appear).

[25] B. Li and K.H. Wang. Nonstop: Continuous multimedia streaming in wireless ad hoc networks with node mobility. *IEEE Journal on Selected Areas in Communication*, 21(10):1627–1641, 2003.

[26] J. Liu and V. Issarny. QoS-aware service location in mobile ad hoc networks. In *Proceedings of the IEEE International Conference on Mobile Data Management*, pages 224–235, 2004.

[27] M. Mamei, F. Zambonelli, and L. Leonardi. Tuples on the air: A middleware for context-aware computing in dynamic networks. In *Proceedings of the International Conference on Distributed Computing Workshops*, pages 342–347, May 2003.

[28] A. L. Murphy, G. P. Picco, and G.-C. Roman. LIME: A middleware for physical and logical mobility. In *Proceedings of the 21st International Conference on Distributed Computing Systems*, pages 524–533, April 2001.

[29] G. P. Picco, A. L. Murphy, and G.-C. Roman. On global virtual data structures. In D. Marinescu and C. Lee, editors, *Process Coordination and Ubiquitous Computing*, pages 11–29, August 2002.

[30] G.-C. Roman, C. Julien, and Q. Huang. Network abstractions for context-aware mobile computing. In *Proceedings of the 24th International Conference on Software Engineering*, pages 363–373, May 2002.

[31] R. Srinivasan and C. Julien. Passive network awareness for adaptive mobile applications. In *Proceedings of the 3rd International Workshop on Managing Ubiquitous Communications and Services*, pages 22–31, 2006.

[32] T. Yu and K.-J. Lin. Service selection algorithms for composing complex services with multiple QoS constraints. In *Proceedings of the International Conference on Service Oriented Computing*, pages 130–143, 2005.

[33] X. Yu. Improving TCP performance over mobile ad hoc networks by exploiting cross-layer information awareness. In *Proceedings of the ACM International Conference on Mobile Computing and Networking*, pages 231–244, 2004.

Adaptive Coordination for Dynamic Mobile Sytems

Principal Investigator Time

The PI will recruit two graduate students to work on this project. These graduate students will be paid through stipends funded by the project, and they will be expected to work on the projects' tasks a minimum of 20 hours per week. In addition to advising these students and working on the project's research in the academic year, the PI will also dedicate the equivalent of 100% of one month of each of the project's summers to full-time work on the project.

Within the Mobile and Pervasive Computing Lab, the PI is currently responsible for advising four PhD students who are funded through the projects listed below. If funded, this project would take over funding of one of these students and add another student to the group.

Current Projects

- *CSR–SMA: ACLI-ware: Dynamic Data-Driven Control for Wirelessly Implemented Application Systems* (7/2006–7/2008): NSF funded, joint with Dr. Sriram Vishwanath; budgeted at $75,000 for two years, 1 RA for two years.
- *Cooperative Communication and Architectures for Cross-Layer Coordination* (7/2006–12/31/2006): DARPA funded, joint with Dr. Scott Nettles; budgeted at $118,401 for CY 2006, 3 RAs for Fall 2006.
- *Computer Science Study Group* (4/2006–3/2007): DARPA funded; budgeted at $87,328 for one year, 3 months summer salary (for Summer 2006) and 1 RA for Fall 2006.
- *SGER: Enabling Truly Pervasive Computing: Communications Meets Software Engineering* (3/2006–2/2008): NSF funded; budgeted at $200,000 for two years; covers 1 1/2 months summer salary (for Summer 2007) and 2 RAs for two years.

Pending Projects

- *CI-TEAM: Educating a Competitive, Cyberinfrastructure-Savvy Engineering and Construction Workforce* (NSF): request is for 1/2 month summer (for summer 2008) and 1 RA for two years (9/1/2006–8/31/2008).
- *NeTS-NBD: Adaptive Application-Centered Communication in Mobile and Pervasive Computing* (NSF): request is for 1/2 month dummer salary (for Summer 2007-2008) and 1 month summer salary (for Summer 2009) and 2 RAs for three years (9/1/2006–8/31/2009).